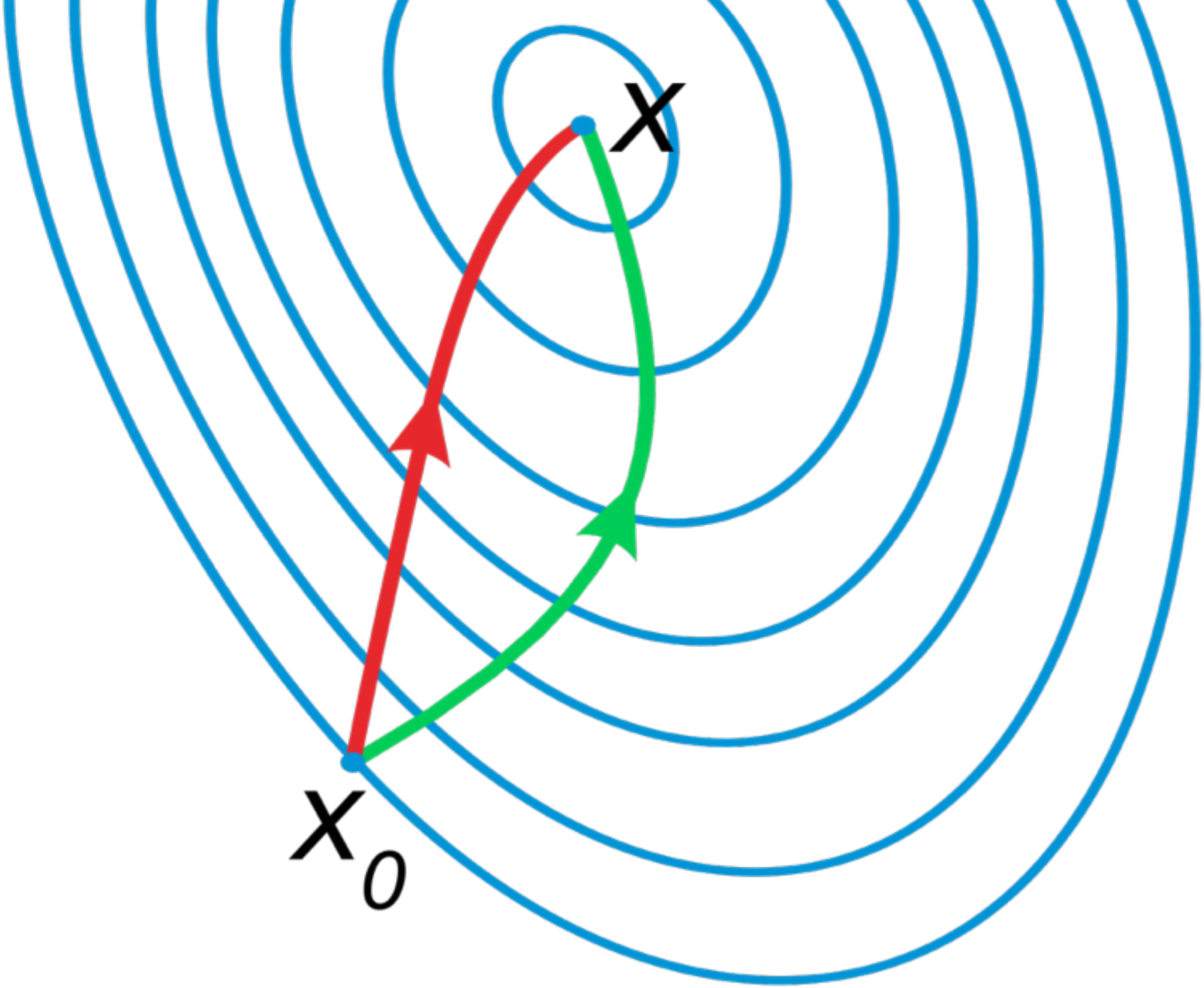


Proximal Algorithms

For distributed and constrained optimization

Niru Maheswaranathan

October 14, 2015

Outline

- Distributed optimization in machine learning
- Dual ascent and augmented Lagrangians
- Proximal Operators and Proximal Algorithms
- Evaluating proximal operators
- Interactive Demo

Outline

- Distributed optimization in machine learning
- Dual ascent and augmented Lagrangians
- Proximal Operators and Proximal Algorithms
- Evaluating proximal operators
- Interactive Demo

Distributed optimization in machine learning

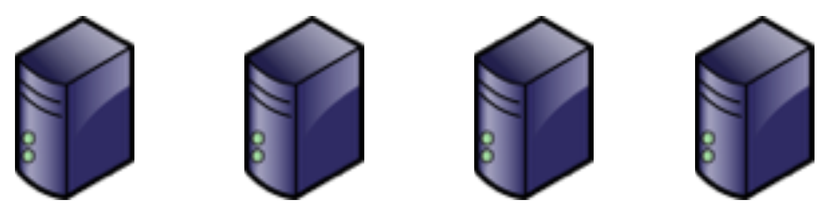
We would like to solve the following:

$$\text{minimize} \quad \sum_i f_i(x)$$

where ***i*** indexes data batches, different loss functions, regularizers, etc.

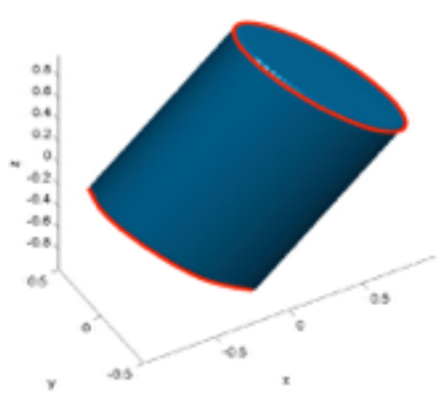
Ideally, we would like to distribute the work amongst many workers

$f_1(x)$ $\cdots$ $f_m(x)$

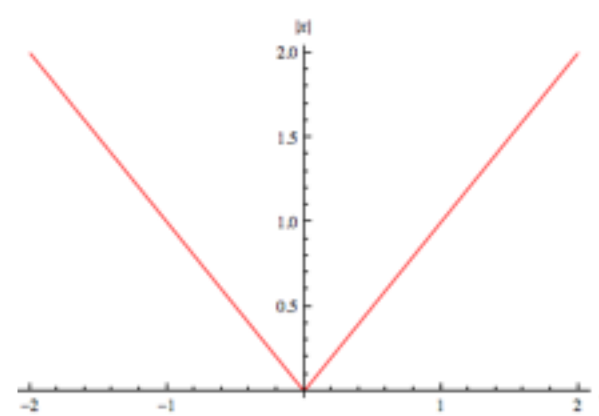


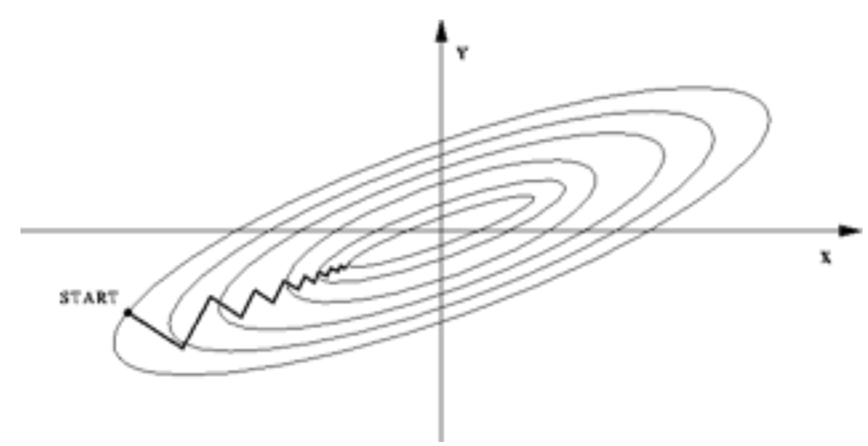
And be able to handle: constraints, non-smooth terms, poor conditioning

$f_i(x) = \mathbf{I}\{x \in \mathcal{C}\}$



$f_i(x) = \lambda \|x\|_1$





Gradient Descent

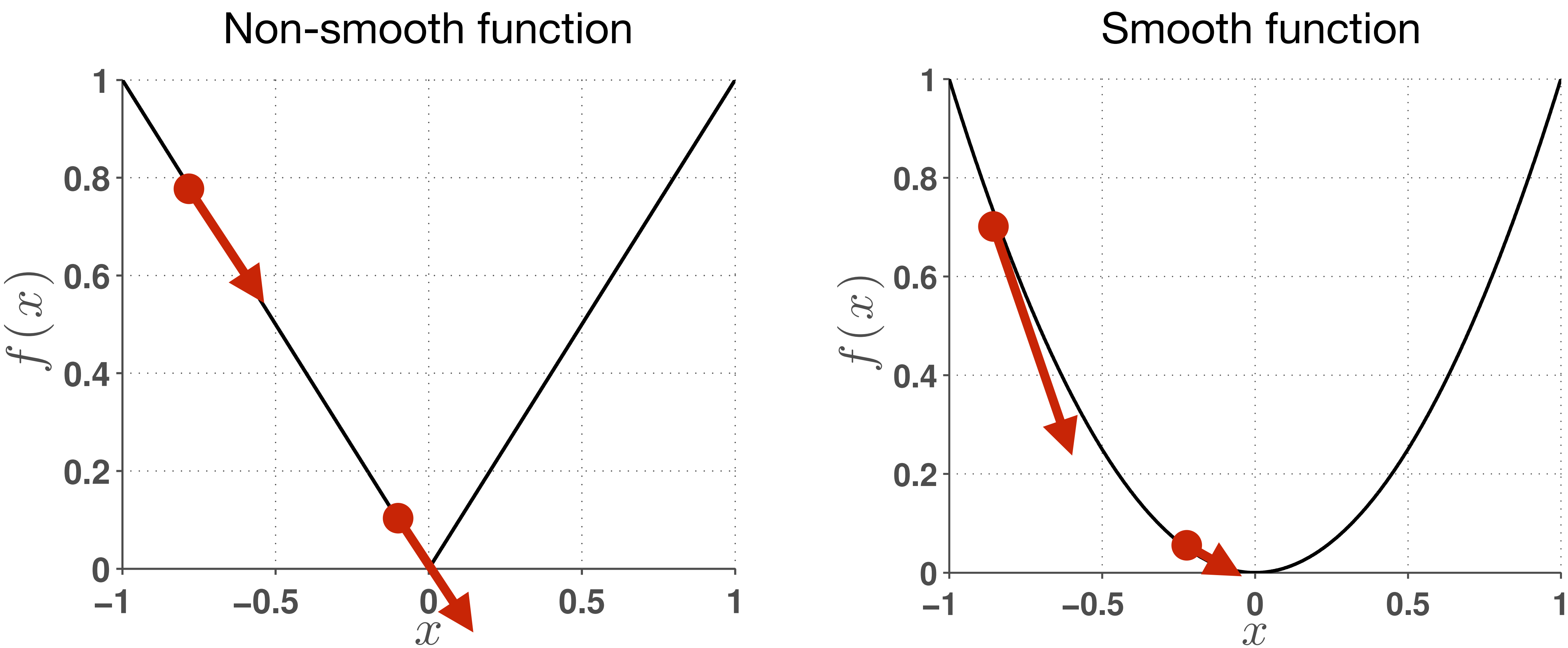
$$\text{minimize} \quad \sum_i f_i(x)$$

Gradient descent update:

$$\boxed{x^{k+1} = x^k - \alpha^k \sum_i \nabla f_i(x)}$$

- [x] Works for smooth, convex functions
- [x] Simple implementation
- [x] Can distribute the gradient computation for each subproblem
- [ ] Constraints? Non-smooth functions? Ill-conditioned problems?

Gradient descent on a non-smooth function



Outline

- Distributed optimization in machine learning
- Dual ascent and augmented Lagrangians
- Proximal Operators and Proximal Algorithms
- Evaluating proximal operators
- Interactive Demo

Dual Ascent

Consider:

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax = b \end{array}$$

Decomposes over *x*

Lagrangian: $$L(x, \lambda) = f(x) + \lambda^T (Ax - b)$$

Dual: $$g(\lambda) = \min_x f(x) + \lambda^T (Ax - b)$$

Dual ascent:

$$\boxed{\begin{aligned} x^{k+1} &= \operatorname*{argmin}_x L(x, \lambda^k) \\ \lambda^{k+1} &= \lambda^k + \alpha(Ax^{k+1} - b) \end{aligned}}$$

- **Brittle; only converges under strong assumptions**
- **Decomposes across subproblems!**

Augmented Lagrange Methods

(also known as: Method of Multipliers)

$$\tilde{L}(x,\lambda) = f(x) + \lambda^T(Ax - b) + \frac{\rho}{2}\|Ax - b\|$$

- 😀 Adds minimum curvature to *f(x)*
- 😞 Not decomposable over *x*

$$\boxed{\begin{aligned} x^{k+1} &= \underset{x}{\operatorname{argmin}}\, \tilde{L}(x,\lambda^k) \\ \lambda^{k+1} &= \lambda^k + \rho(Ax^{k+1} - b) \end{aligned}}$$

* **Converges under weak assumptions!**
* **But we lose decomposability :(**

Alternating Direction Method of Multipliers (ADMM)

$$
\begin{aligned}
\text{minimize} \quad & f(x) + g(z) \\
\text{subject to} \quad & x - z = 0
\end{aligned}
$$

Augmented Lagrangian:

$$\tilde{L}(x, z, \lambda) = f(x) + g(z) + \lambda^T (x - z) + \frac{\rho}{2} \|x - z\|_2^2$$

ADMM:

$$
\begin{aligned}
x^{k+1} &= \operatorname*{argmin}_x \tilde{L}(x, z^k, \lambda^k) \\
z^{k+1} &= \operatorname*{argmin}_x \tilde{L}(x^{k+1}, z, \lambda^k) \\
\lambda^{k+1} &= \lambda^k + \rho(x^{k+1} - z^{k+1})
\end{aligned}
$$

* **Converges under weak assumptions!**
* **And it is easily parallelizable! (as we will see…)**

Scaled form of ADMM

$$x^{k+1} = \underset{x}{\operatorname{argmin}}\, \tilde{L}(x, z^k, \lambda^k)$$

$$\tilde{L}(x, z, \lambda) = f(x) + g(z) + \lambda^T(x - z) + \frac{\rho}{2}\|x - z\|_2^2$$

(pull the linear term inside the quadratic)

Scaled form:

$$\tilde{L}(x, z, \lambda) = f(x) + g(z) + \frac{\rho}{2}\|x - z + u\|_2^2 + \text{const.}$$

(where $u = \lambda/\rho$)

These are *proximal operators*

ADMM
(scaled form)

$$\boxed{\begin{aligned} x^{k+1} &= \underset{x}{\operatorname{argmin}}\, f(x) + \frac{\rho}{2}\|x - z^k + u^k\|_2^2 \\ z^{k+1} &= \underset{z}{\operatorname{argmin}}\, g(z) + \frac{\rho}{2}\|x^{k+1} - z + u^k\|_2^2 \\ u^{k+1} &= u^k + x^{k+1} - z^{k+1} \end{aligned}}$$

Outline

- Distributed optimization in machine learning
- Dual ascent and augmented Lagrangians
- Proximal Operators and Proximal Algorithms
- Evaluating proximal operators
- Interactive Demo

Proximal operators

Definition

$$\mathrm{prox}_f(v) = \mathrm{argmin}_x \left(f(x) + \frac{1}{2}\|x - v\|_2^2 \right)$$

Minimize

$$\boxed{f(x)} + \boxed{\frac{1}{2}\|x - v\|_2^2}$$

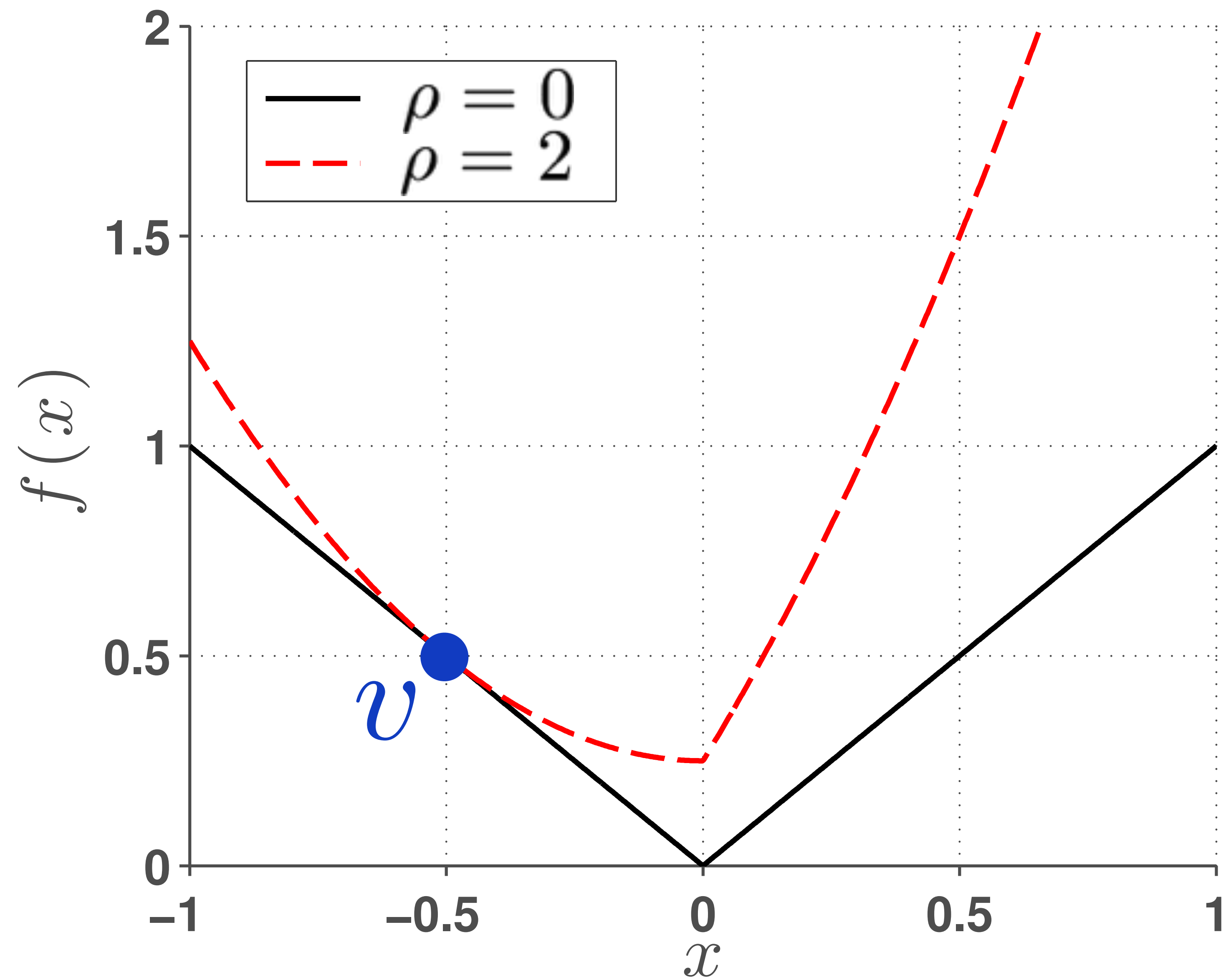
Trade-off:

Make this function / penalty small

but stay close to the point *v*

Proximal operator for the l1-norm

$$f(x) + \frac{\rho}{2}\|x - v\|^2$$



Interpretations

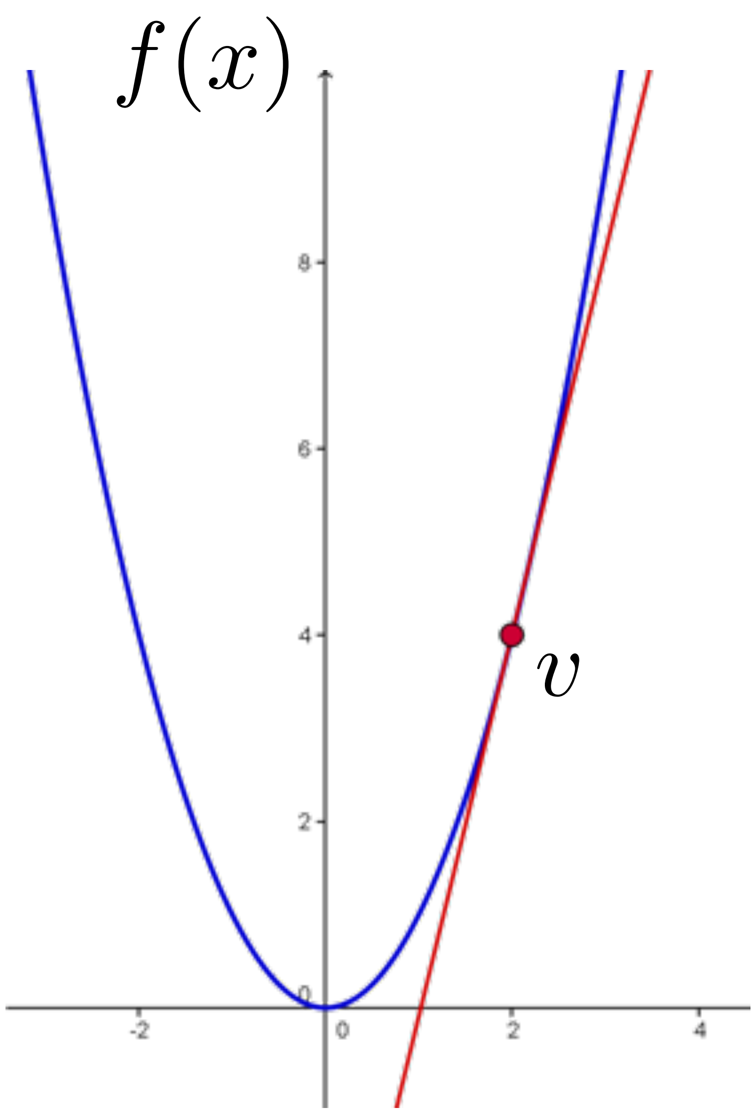
$$\text{prox}_{\rho,f}(v) = \underset{x}{\text{argmin}} \left(f(x) + \frac{\rho}{2} \|x - v\|_2^2 \right)$$

1. Function smoothing
   - The added term *smooths* out the function f(x) locally near *v*

2. MAP estimation with a Gaussian prior
   - Think of f(x) as a negative log-likelihood term
   - The added term is the negative log-probability for a Gaussian prior centered at *v* with identity covariance

3. Connection to gradient methods

Aside: proximal operators and gradient descent



First-order approximation to the function near the point *v*:

$$\hat{f}(x) \approx f(v) + \nabla f(v)^T (v - x)$$

Let's apply the proximal operator to this approximation:

$$\mathrm{prox}_f(x) = \underset{x}{\mathrm{argmin}} \left(\hat{f}(x) + \frac{1}{\alpha} \|x - v\|^2 \right)$$

$$\underset{x}{\mathrm{argmin}}\, f(v) + \nabla f(v)^T (v - x) + \frac{1}{\alpha} \|x - v\|^2 v$$

Analytic minimization:

$$\nabla \hat{f}(x) = \nabla f(v) + \frac{1}{\alpha}(x - v) = 0$$

$$x = v - \alpha \nabla f(v)$$

yields the gradient descent update equation

Proximal algorithms (recap)

Problem: $\text{minimize} \quad f(x) + g(x)$

Operator splitting:

$$\begin{array}{ll} \text{minimize} & f(x) + g(z) \\ \text{subject to} & x = z \end{array}$$

Algorithm (ADMM):

$$\begin{aligned} x^{k+1} &:= \mathbf{prox}_{\lambda f}(z^k - u^k) \\ z^{k+1} &:= \mathbf{prox}_{\lambda g}(x^{k+1} + u^k) \\ u^{k+1} &:= u^k + x^{k+1} - z^{k+1}, \end{aligned}$$

We benefit if proximal operators for f(x) and g(x) are easy to evaluate, but the operator for f+g is difficult to evaluate

Proximal consensus

Problem: $$\text{minimize} \quad \sum_i f_i(x)$$

Algorithm:

$$x_i^{k+1} = \underset{x}{\operatorname{argmin}} \left(f_i(x) + \frac{\rho}{2} \|x_i - \bar{x}^k + u_i^k\|_2^2 \right)$$

$$\bar{x}^{k+1} = \frac{1}{m} \sum_{i=1}^{m} x_i^{k+1}$$

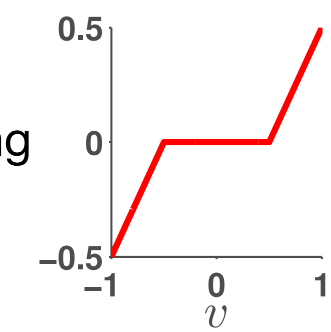
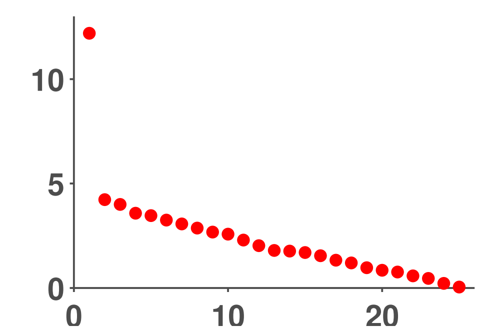
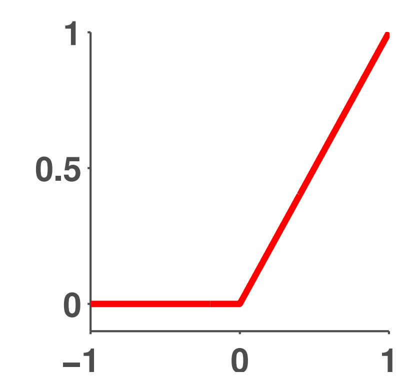
$$u_i^{k+1} = u_i^k + (x_i^{k+1} - \bar{x}_i^{k+1})$$

- ☑ Works for smooth, non-smooth, Ill-conditioned convex functions
- ☑ Can run the proximal update for each subproblem in parallel
- ☑ Guaranteed to converge, finds reasonable solutions quickly (similar to first order methods)

Outline

- Distributed optimization in machine learning
- Dual ascent and augmented Lagrangians
- Proximal Operators and Proximal Algorithms
- Evaluating proximal operators
- Interactive Demo

Evaluating proximal operators

Property	$f(x)$	$\mathrm{prox}_f(v)$
Least squares	$\frac{1}{2}\|Ax-b\|_2^2$	$(A+\rho I)^{-1}(\rho v - b)$
Sparsity (l1-norm)	$\|x\|_1$	Soft thresholding 
Low-rank matrix (nuclear norm)	$\|X\|_*$	Soft thresholding the singular values 
Non-negative values	$\mathbf{I}\{x \geq 0\}$	Clip negative values 

Outline

- Distributed optimization in machine learning
- Dual ascent and augmented Lagrangians
- Proximal Operators and Proximal Algorithms
- Evaluating proximal operators
- Interactive Demo

Demo

https://github.com/nirum/descent-tutorial

References

- N. Parikh and S. Boyd, *Proximal Algorithms*, 2014. http://stanford.edu/~boyd/papers/pdf/prox_algs.pdf
- S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*, 2011. http://stanford.edu/~boyd/papers/pdf/admm_distr_stats.pdf